**Music Genre Classification**

**PROJECT REPORT**

**CONTENTS**

1. Introduction
   1. Overview
   2. Purpose
2. Literature Survey
   1. Existing problem
   2. Proposed Solution
3. Theoretical Analysis
   1. Block diagram
   2. Hardware/Software designing
4. EXPERIMENTAL INVESTIGATIONS
5. FLOWCHART
6. RESULT
7. ADVANTAGES & DISADVANTAGES
8. APPLICATIONS
9. CONCLUSION
10. FUTURE SCOPE
11. BIBLIOGRAPHY

**APPENDIX**

Source code

Introduction

* 1. Overview

Ever wanted an amazing playlist that includes songs you like, as well as introducing you to similar songs that you may have never listened to before? This is a million-dollar question faced by music streaming services like Apple Music, Pandora, Spotify, and YouTube, to name a few. As engineers we view a song as, depending on length, approximately million data-points. In order to classify songs as similar we need to look at these data-points and it becomes increasingly difficult to do this with the raw data especially when new songs are introduced every day. Also, similarity between songs is a difficult thing to define as the parameters used to describe how similar two songs are inherently subjective and cannot be easily translated into an algorithm. A genre is defined as a category of artistic composition characterized by similarities in form, style or subject matter. One of the ways we can say two tracks are similar is by looking at the genre they belong to. Two songs belonging to the same genre are usually going to have more similarities than two songs belonging to different genres. The classification based on genre can be done by extracting information that can be retrieved from the raw data.

2. Literature Survey

2.1 Existing problem

The problem with this approach is the subjectivity and ambiguity of the categorization used for training and validation. Often genres don’t even correspond to the sound of the music but to the time and place where the music came up or the culture of the musicians creating it. Some authors try to explain the low performance of their classification methods by the fuzzy and overlapping nature of genres. An analysis of musical similarity showed bad correspondence with genres, again explained by their inconsistency and ambiguity.

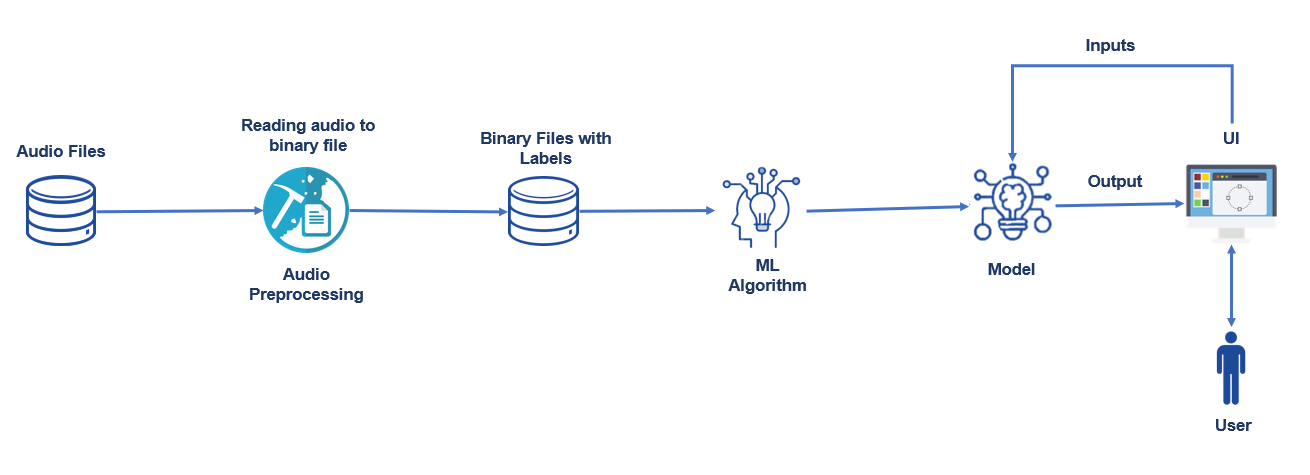
2.2 Proposed Solution

We used a dataset that was was taken from the Marsyas software it is open source and open to anyone to use, in particular in academia, but also in the industry. Our initial data consisted of 1000 .au files (a file extension used for audio files), split in 10 different music genres (Disco, Metal, Pop, Hip Hop, Rock, Blues, Classical, Country, Jazz, Reggae), with 100 samples of 30 seconds for each genre. Our next step was to find a way to convert this raw data to a format that our machine learning techniques could use. We were referred to the use of Mel Frequency Cepstral Coefficients. Combining SoX together with James Lyons’ MFCC software to obtain our coefficients. In addition, the delta and acceleration values of the MFCCs at each time step, calculated as ∆vi = vi − vi−1 and acc(vi) = ∆vi − ∆vi−1 were used as features. These new features are extremely important as they measure the transitions between time steps. Therefore, each training example was represented as a large matrix of roughly 3000 rows (rows correspond to time steps) and 39 columns. Now, most of the algorithms that we used usually treat vectors as inputs, hence we either applied PCA to our matrix, or we flattened the matrix to an extremely large vector, and then used this structure as a training example. Below is a visualization of what a subset of our dataset looks like after applying PCA to reduce the input to three dimensions:

As a slight divergence, we also investigated the topic of composer classification, this time using a different format of files, and hence different features. In order to do this MIDI files were used. The MIDI format contains all the information pertaining to the partition of a piece, which is mainly note pitches and note durations. We used the Python library music21 to process the raw MIDI files. We chose to isolate the melody which we approximated by the uppermost sequence of notes of a song and we set all the note durations to 1. Hence, a song was there-fore reduced to a sequence of pitches with no timing.

3.Theoretical Analysis

3.1 Block diagram



3.2 Software designing

One key feature of these services is the playlists, often grouped by genre. This data could come from manual labelling by the people publishing the songs. But this does not scale well and might be gamed by artists who want to capitalize on the popularity of a specific genre. A better option is to rely on automated music genre classification. In particular, we evaluated the performance of standard machine learning vs. deep learning approaches. What we found is that feature engineering is crucial, and that domain knowledge can really boost performance.

After describing the data source used, I give a brief overview of the methods we used and their results. In the last part of this article, I will spend more time explaining the way the TensorFlow framework in Google Colab can perform these tasks efficiently with GPU or TPU runtimes thanks to the TF Record format.

4. EXPERIMENTAL INVESTIGATIONS

It is defined as the percentage of correctly classified test labels. Table 2 provides the accuracy of the classifiers detailed in section

Classifiers Training accuracy

|  |  |  |
| --- | --- | --- |
| KNN | 68% | 62% |
| SVM | 61% | 62% |
| Decision Tree | 60% | 47.6% |
| Random Forest | 77.6% | 58.8% |
| Deep Neural Network with dropout | 80% | 71% |

Comparing the training and validation accuracies of various classifiers used.

Audio data is becoming an important part of machine learning. We are using audio to interact with smart agents like Siri and Alexa. Audio will also be important for self-driving cars so they can not only “see” their surroundings but “hear” them as well. I wanted to explore deep learning techniques on audio files and music analysis seems to be an interesting area with lots of promising research. In this blog I have looked at architectures that combine CNNs and RNNs to classify music clips into 8 different genres. I have also visualized filter activations in different CNN layers. If you are new to deep learning and want to learn about CNNs and deep learning for computer vision.

# Data Set and Conversion to Mel-Spectograms

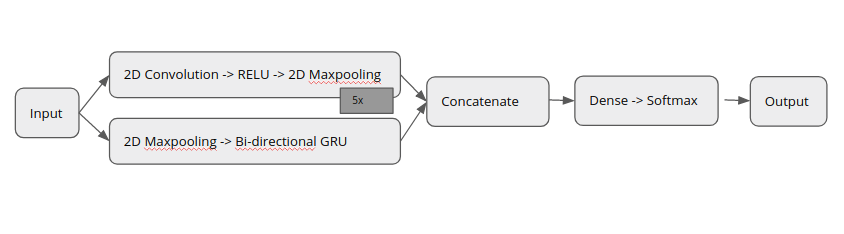
There are a few different datasets with music data — [GTZan](http://marsyasweb.appspot.com/download/data_sets/) and Million Songs data set (MSD) are 2 of the ones most commonly used. But both of these data sets have limitations. GTZan only has 100 songs per genre and MSD has well 1 million songs but only their metadata, no audio files. I decided to use the free music archive small dataset. You can use their github link to download the small dataset (8 GB) which has raw audio files + metadata. The FMA small data set that I used had 8 genres and 1000 songs per genre evenly distributed. The eight genres are Electronic, Experimental, Folk, Hip-Hop, Instrumental, International, Pop and Rock.

**Converting audio data into mel-spectogram**

Each audio file was converted into a spectogram which is a visual representation of spectrum of frequencies over time. A regular spectogram is squared magnitude of the short term Fourier transform (STFT) of the audio signal. This regular spectogram is squashed using mel scale to convert the  
audio frequencies into something a human is more able to understand. I used the built in function in the librosa library to convert the audio file directly into a mel-spectogram. The most important parameters used in the transformation are — window length which indicates the window of time to perform Fourier Transform on and hop length which is the number of samples between successive frames. The typical window length for this transformation is 2048 which converts to about 10ms, the shortest reasonable period a human ear can distinguish. I chose hop length of 512. Further more the Mel-spectrograms produced by Librosa were scaled by a log function.

## **Parallel CNN-RNN Model**

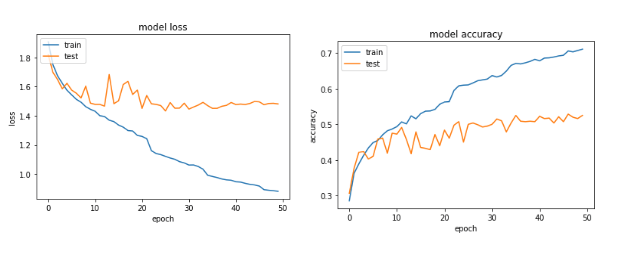
The key idea behind this network is that even though CRNN has RNNs to be the temporal summarizer, it can only summarize temporal information from the output of CNNs. The temporal relationships of original musical signals are not preserved during operations with CNNs. This model passes the input spectogram through both CNN and RNN layers in parallel, concatenating their output and then sending this through a dense layer with soft max activation to perform classification as shown below.



Parallel CNN-RNN Model

The convolutional block of the model consists of 2D convolution layer followed by a 2D Max pooling layer. This is in contrast to the CRNN model that uses 1D convolution and max pooling layers. There are 5 blocks of Convolution Max pooling layers. The recurrent block starts with 2D max pooling layer of pool size 4,2 to reduce the size of the spectogram before LSTM operation. This feature reduction was done primarily to speed up processing. The reduced image is sent to a bidirectional GRU with 64 units.  
The outputs from the convolutional and recurrent blocks are then concatenated resulting in a tensor of shape. Finally we have a dense layer with SoftMax activation.  
The model was trained using RMSProp optimizer with a learning rate of 0.0005 and the loss function was categorical cross entropy. The model was trained for 50 epochs and Learning Rate was reduced if the validation accuracy plateaued for at least 10 epochs.

Figure below shows the loss and accuracy curves from this model



This model had a validation loss of around 51%. Both models have very similar overall accuracies which is quite interesting but their class wise performance is very different. Parallel CNN-RNN model has a better performance for Experimental, Folk, Hip-Hop and Instrumental genres. The ensembling of both these models should produce even better results.

K- Nearest Neighbors-KNN:

K- Nearest Neighbors-KNN: KNN is one of the

KNN is one of the distance based supervised learning algorithms.

When solving the classification problem with this

method, a model is not created and the test

operation is performed on the labelled samples in

the data set. A new instance of the class label will

be calculated from the distance from the instances

in the dataset. From these calculated distances, the

class tag is estimated by voting on the class labels

of the nearest k. When calculating the distance, the

Euclidean, Manhattan distance formulas are often used.

Naive Bayes-NB:

Naïve Bayes-NB: The Naive Bayes algorithm is a

The Naive Bayes algorithm is a probabilistic supervised learning algorithm that

generates a classification model by calculating the

preliminary probabilities from the data in the data

set and classifies the new data according to this

model. It is an algorithm that can be used in various

problems because it is compatible with every kind

of data and simple statistical calculations are

required.

• Decision Tree-DT: Decision trees are learning

algorithms that provide a supervised and model-

based approach. It tries to identify the most

distinctive feature in the data set as the root node of

the tree. An entropy calculation is made when the

most distinguishing feature is found. There are also

different metrics in the literature that provide

differentiating features.

• Support Vector Machine-SVM: SVM is one of

model based supervised learning algorithms. DVM

is based on the principle of training for a decision

surface that will allow the two classes to

distinguish one another. This decision surface is

created by optimizing the boundary regions of the

two classes. SVM can be used in multi-class data

sets other than two-class data sets.

• Random Forest-RF: Random Forest (RF) is also

utilized to the same feature set to search the success

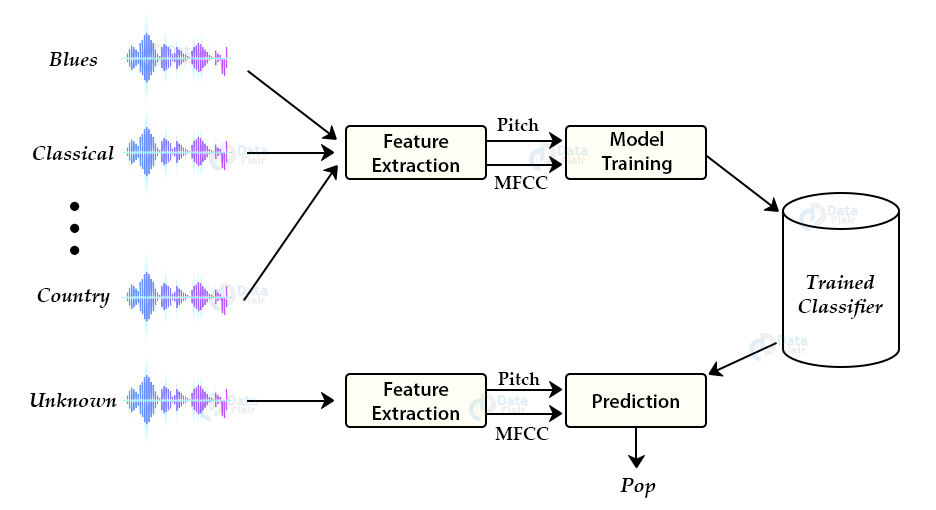
of an ensemble technique as to the music genre

classification. RF can be used as a combination of

multiple decision trees with bagging sample

selection strategy.

5. Flow chart



6.RESULT

In our study, the Deep Neural Network performs best as it has the highest training (80%) and validation (71%) accuracy. While the decision tree classifier performs the worst with the lowest accuracy due to its instability with large data. It is evident that SVM with RBF kernel outperforms decision tree. KNN is a widely used supervised learning classifier and it's easy to implement. KNN performs better than SVM and decision tree in our study. While a Random Forest classifier yields a far better training accuracy but it fails to classify the test samples correctly. We can analyze which features play a vital role during prediction of genre, in the classification task. To do this analysis, we have ranked the top 25 features that are used to predict the genre of music. As shown in figure 4, the ‘root mean square energy (rmse)’, ‘chroma\_shift’ and ‘mel frequency cepstral coefficients 4 (mfcc4)’ play a significant role in the music genre classification task. A previous study has shown that ‘rmse’ plays an important role in the music genre classification.

7.ADVANTAGES & DISADVANTAGES

There are pros and cons for these two methods. Both methods are not able to be implemented in real time since we need to process the information of whole piece of music. And theoretically, the complexity of the decision tree is lighter since we could store the pretrained model. And the complexity of the KNN grows quadratically with the window size (Tw), linearly with the signal length (N) and its efficiency is worse when the data set is large. If we have a relatively small data set with a large number of classes, it would be suggested to use KNN. Otherwise, the decision tree is preferred. For both methods, the performance of classifying Jazz and Disco genre music is not good enough, whose accuracy is around 50% to 60%.

Several potential improving methods are discussed below.

Add more specific audio features. In this project, we use several spectral analysis techniques to extract the song features. For example, we calculate spectral flux for each song, which represents the rate of change of spectral amplitude and thus indicates the intensity of the music. This feature may work well for classifying classical and pop music. However, it may perform poorly for classifying pop and disco music’s because the intensity of music is not specific enough for classification. We need to find out more specific audio features, like chord features.

Construct audio features with proper dimension. One problem arises from adding more specific audio features is the size of extracted features. Adding more specific audio features will lead to larger and more complex inputs for further learning methods, which will become a large overhead of computational cost. A good trade-off between extracted features and computational cost should be further studied.

Prune and add weights to audio features. In our study, we found that the simple combination of different features will not certainly improve the performance, although the separate feature will lead to better performance. For example, combining LPC with spectral flux and combining LPC with spectral roll off will both give a good result. However, combining LPC and spectral flux with spectral roll off will otherwise hurt the performance. Therefore, it should be careful when choosing possible combination of different features.

On one hand, it's better to combine features from different perspectives and prune unnecessary audio features to reduce computational overhead. For example, we do not use MFCC in decision tree model since LPC is already used to filter out noise and transmit spectral envelope information. On the other hand, we can add weights to different audio features for further learning methods, thus improving the classification accuracy between similar genres like pop and disco.

Increase the size of dataset. In practical application of machine

8.APPLICATIONS

Music genre classification forms the basis steps for any music recommendation system.

With the advent of Deep neural networks, CNN and CRNN networks have found applications not only in music generation but also efficient music genre classification.

1.By using this Music Genre Classification users can easily listen to the music that they are willing to listen.

2. Music Genre Classification classifies the types of music and sets all the type of same music in single folder.

3.Searching for song they require will be less as the type of song is already classified.

4.Code efficiency will become more strength as the user uses it more.

5.Less time consumption for searching of song.

EX: Spotify, SoundCloud.

9.CONCLUSION

The classifier that works best is SVM with Polynomial Kernel. Some classifier are very efficient for some specific genres (like SVM with RBF Kernel for“country”). The highest verified accuracy on the GTZAN dataset is reported at approximately 84%. The ensemble method improves upon Polynomial SVM classifier by reducing large errors in classifying specific genres.

Two state of the art approaches are proposed. First, an ensemble of XGBoost model trained on handcrafted time and frequency domain features with BERT network trained on lyrics of 5000 songs. Second, the multi modal fusion network which uses KQV attention and multiple modality fusion for classification of genres. The dense co attention model is found to be the best performing network among all the other networks.

The data collected for multi-frame approach provide a new dimension towards training more robust networks by generating more training data with a smaller number of audio files. Thus, the deep learning models outperform the feature engineering-based models. In the future, a larger dataset can be used and the noise from the data obtained from online resources can be removed before frame processing. The genre prediction network can be used in conjunction with recommender systems to create user centred playlists.

10. Future Scope

In the analysis performed, the overall correctness of classification was higher in almost each case of the mixed VoP in comparison to the Original signal. Also, it was observed that the specific mix of signals improved the correctness of classification of genres where this signal played an important part. This means that for genres where harmonic instruments play an important part, e.g. New Age, Pop, Latin Music, the correctness of classification increased. The same tendency was observed for other mixed VoPs: OD signal for Alternative Rock, Hard Rock & Metal, as well as DanceDj, and New Age. In the case of the OP signal, the improvement in classification of Blues, Classical and New Age was also visible. Overall, a decrease in misclassification between the similar, as well as opposite genres was obtained. In the process of the analysis over 8.000 music tracks, representing 13 music genres, were extracted from the Synat database. Although many research works were published in the area of music genre classification, most of them, with some exceptions, analyse only a few genres represented by ∼ 1.000 songs in total.

1.Clubbing of two genres will be included.

2.More clearly classification is done.

3.Will become more user friendly.

4.Multiple genre mix and their classification will be included.

11.BIBLIOGRAPHY

* Convolutional Recurrent Neural Networks for Music Classification
* Music Genre Classification with paralleling Recurrent Convolutional Neural Network.

Appendix

Source Code

https://github.com/SAISREES/Music-Genre-Classification.git

Team Members

1.Kavya Rayala(17RH1A05K2)

2.Sai Sree S(17RH1A05K8)

3.Sravani Pulusu(17RH1A05J6)

ic genre classiﬁcation has been a wid-

ied area of research since the early days of the

Internet. Tzanetakis and Cook (2002) addressed

this problem with supervised machine learning ap-

proaches such as Gaussian Mixture model and k-

nearest neighbour classiﬁers. They introduced 3

sets of features for this task categorized as tim-

bral structure, rhythmic content and pitch con-

tent. Hidden Markov Models (HMMs), which

have been extensively used for speech recognition

tasks, have also been explored for music genre

classiﬁcation (Scaringella and Zoia,2005;Soltau

et al.,1998). Support vector machines